OAuth stands for Open Authorization. It should probably be called OAuthz to remind people that OAuth should be used for authorization and not authentication. It is generally used as a mechanism to delegate authorization. An end user can allow a party access to its data which is stored at another party.

The problem that OAuth tries to solve, in a nutshell, is this: a user wants to share its data between two applications. The crazy stupid one to do this would simply be to give his/her password to the application that he/she wants to share data with. And the application can simply log in and retreive the data. This would of course be crazy, because everyone would have to share their creds to a lot of applications.

So how can we share data between applications without sharing the users credentials?

Let's create some better terminology for this: 1. Resource owner (RO) - usually the enduser. 2. Client - The application that wants access to the Resources owners resources. 3. Authorization server - The server that asks the RO for access on behalf of the client. 4. Resource Server - Server that stores the ROs data.

It is common that the Authorization server and the resource server is the same actor. Facebook, Google, Github.

In OAuth terminology one talks about different grant_types. There are four different grant_types: 1. Authorization code 2. Implicit 3. Resource owner password credentials - This is the problem that I previously wrote that OAuth tries to solve. 4. Client credentials

Authroization code - Code Flow

Let's walk through the grant type Authorization code. This is usually called Code Flow. It is the most complex form, but probably also the most secure.

It is divided into two parts: Authorization flow and Token Flow.

OAuth Flow Authorization flow - runs in the browser 1. User enters Client website/app. 2. User is redirected to Authorize endpoint of OAuth server. 3. OAuth Server redirect user to Authentication Server, since the user is not authenticated. 4. User authenticats at Authentication server, and is redirected back to OAuth server. 5. OAuth server issues a one-time token (Authorization Code)

Token Flow - Runs on the backend 6. The Client post a request to the /token endpoint with the Authorization code and the client credentials (Client ID and Client Secret). 7. The Auth Server returns a access token and a refresh token.

Common attacks

Token / Code stealing

CSRF - Missing State param

Token impersonation